Strategic Evolution of ESE Data Systems (SEEDS)
Survey of Cost Estimation Tools

Deliverable 34.05.01, November 30, 2001

Introduction

The study team[1] has completed a survey of commercial tools for estimating costs of hardware and software development for data systems. An effort was made to conduct this study with an eye toward the next step — to develop a model for estimating the lifecycle costs of evolving Earth Science Enterprise (ESE) data systems. This report summarizes the results of the survey and delineates the availability and usefulness of existing costs estimation tools.

The result of research efforts into software engineering and cost estimation methodologies can be summarized in a few statements[2] (see inset) that are independent of methodology and have survived over time. Some researchers observe that, in practice, the most common method is cost estimation by analogy. Meanwhile, cost estimation tools abound, each with its own set of pluses and minuses, and no tool stands out as the single best answer. Software cost estimation tools employ one or more of several known methods: parametric modeling, knowledge-based modeling, rule induction, fuzzy logic, dynamic modeling, neural networks, or case-based reasoning. Current research efforts into cost modeling focus on the latter two methods plus a few architectural approaches and the COCOMO Project.

Lessons Learned

- There is a minimum development time below which a system cannot be completed successfully.
- There is a useful trade-off between time and effort.
- There is a functional coupling between size, schedule, effort, and reliability—change one, the others all change.
- There is great payoff from improving the productivity of the development process.
- There is no silver bullet.
- New development methods change the way you size a project.

Summary of Existing Cost Estimation Tools

The survey identified 21 commercial-off-the-shelf (COTS) cost estimation tools. This set reflects a wide range of methodologies, levels of expertise or maturity, features, and cost. Most of the tools are parametric models. Some tools address hardware as well as software, but most do not. A few tools offer a stochastic model. Some tools are available for download over the Internet free of charge. Table 1 summarizes each software tool (both empirical and stochastic models) for estimating software data system development costs and notes certain relevant features.

Developers of parametric models derive Cost Estimating Relationships (CERs) by regression analysis on historical data on project attributes (cost drivers) and cost. Cost estimation models use these relationships as scale factors in an exponential equation to calculate the effort and schedule required for the software development effort. Each parametric cost estimation model in this survey is either a variant of the Constructive Cost Model (COCOMO), Putnam manpower build-up model, or some proprietary (homegrown) formulation.

[1] NewDISS Requirements and Cost Estimation, ESDIS Task 34, NAS5-00154.
[2] Larry Putnam, Key Things We Have Learned, white paper.

Table 1. Cost Estimation Tools

Product	Description	Features
ACEIT	Automated Cost Estimating Integrated Tools (ACEIT) from Tecolote Research is an automated architecture and framework for cost estimating and many other analysis tasks. ACEIT is a government-developed tool that has been used for over a decade to standardize the Life Cycle Cost estimating process in the government environment. ACEIT is installed at over 400 sites within government and industry. http://www.aceit.com/	• Parametric • Life Cycle • Hardware
ANGEL	Empirical Software Engineering Research Group (ESERG) at Bournemouth University has a research project focused on estimating software development costs using case-based reasoning (analogy). A brief bibliography and the downloadable ANGEL tool are provided. The tool is not well supported. http://dec.bmth.ac.uk/ESERG/ANGEL/	• Analogy
COCOMO Interactive	COCOMO Interactive (Texas A&M University) is an on-line, interactive software package that assists in budgetary and schedule estimation of a software project. This was a class project that is not being supported. http://www.cs.tamu.edu/projects/spring95/431a-docs/intro.html	• COCOMO 81
COCOMO Project	The COCOMO Project is a program of research conducted by Barry Boehm. COCOMO II is an update of COCOMO 1981, which addresses 1990s and 2000s software development practices. A public version of COCOMO II, including a COTS version (COCOTS), is available. USC-CSE, UC Irvine, and 29 affiliate organizations are developing it. http://sunset.usc.edu/research/COCOMOII/index.html	• Parametric • COCOMO II • COTS • Reuse
CoCoPro	CoCoPro from ICONIX Software Engineering is one an integrated suite of 10 analysis and design tools supporting the major phases of the system development lifecycle, including analysis, design, coding and the management of complex systems. It based upon Barry Boehm's constructive cost modeling methods. CoCoPro supports the Intermediate level of the COCOMO methodology. http://www.iconixsw.com/Spec_Sheets/CoCoPro.html	• Parametric • COCOMO Intermediate
Construx Estimate	Construx Software Builders provide Construx Estimate, a free estimation tool that includes the functionality of both COCOMO II and SLIM (the QSM product below). Construx Estimate uses Monte Carlo simulations to model complex interactions in the face of uncertain estimating assumptions, making the company one of the few who offer a stochastic method. http://www.construx.com/estimate/	• Parametric • Stochastic • COCOMO II
COOLSoft	COOLSoft from Wright Williams & Kelly utilizes a hybrid approach of intermediate and detailed versions of COCOMO. This allows for the reuse of existing code, development of new code, and integration of both hardware and third party code. The model comes in a standard Microsoft Excel spreadsheet format. http://www.wwk.com/coolsoft.html	• Parametric • COCOMO • COTS • Reuse
COSMOS	COSMOS (Design Studio Group/Oak Ridge National Laboratory) is unique in that it combines the well-known Function Point and COCOMO models as well as a Rayleigh model of manpower buildup proposed by Larry Putnam. These models can be used independently or work together. http://www-cs.etsu.edu/cosmos/	• COCOMO • Parametric • Function Point • Putnam Method

Product	Description	Features
Costar	Costar from Softstar Systems is a cost estimation tool that supports COCOMO II, COCOMO 81, REVIC, Ada COCOMO, and Incremental COCOMO. Costar is an interactive tool that permits managers to make trade-offs and what-if analyses to arrive at the optimal project plan. Costar 6.0 has 13 built-in models. http://www.softstarsystems.com/	• Parametric • COCOMO II • Incremental COCOMO • Database
Cost Xpert	Cost Xpert (The Cost Xpert Group) is a software cost estimating tool that integrates multiple estimating models into one tool to provide accurate and comprehensive estimates. It claims to be the only tool offering support for sophisticated modeling techniques such as system dynamic modeling, knowledge based modeling, both stochastic and deterministic modeling, and a variety of cost models including the latest release of COCOMO II. http://www.costxpert.com/	• Parametric • Stochastic • System Dynamic • Knowledge-based • Database
ECOM	ECOM (ESA's COst Model) is not a modeling tool. It is a software tool for collection, retrieval and processing of cost data from past ESA programs and projects. ECOM is linked to a hybrid cost estimation tool combining items from the ECOM database, Cost Estimating Relationships (CERs) and estimates using the PRICE-H parametric tool. The combining tool is ACEIT (from Tecolote Research, see above), which has been customized to a European and ESA like environment. http://www.estec.esa.nl/eawww/ecom/ecom.htm	• Database
ESC Open Model	The ESC Open Model (Tecolote Research) is a suite of Cost Estimating Relationships and metrics used to estimate the effort required for commercial off-the-shelf (COTS) and non-developmental item (NDI)-intensive software efforts. http://www.tecolote.com/Services/CostResearch.htm - ESC Open Model	• CER • COTS • Reuse
ESTIMATE Pro	ESTIMATE Professional (Software Productivity Center) makes use of three mature estimation approaches: Putnam methodology, COCOMO II, and Monte Carlo simulation. Putnam methodology is based on the insight that efficiently run software projects follow well-defined patterns that can be modeled with a set of exponential equations. COCOMO II is a continuation of work begun by Barry Boehm. Monte Carlo simulation models complex interactions in the face of uncertain estimating assumptions. http://www.spc.ca/products/estimate/index.htm	• Parametric • COCOMO II • Putnam Method • Monte Carlo Simulation
ParaModel	ParaModel from Mainstay is not database driven, it is a parametric estimating tool. It integrates hardware and software components to create a complete program estimate. ParaModel combines hardware and software estimates, supports any level of the WBS, and provides presentations meaningful to management. http://www.mainstay.com/products/index.html	• Parametric • Hardware
PRICE-S	PRICE-S (Parametric Review of Information for Cost & Evaluation—Software) and PRICE-H from PRICE Systems at Lockheed Martin are well-known cost estimating models for software and hardware. PRICE-H, useful for estimating the cost of hardware development, has limited usefulness in estimating the cost of hardware purchased for use in data systems development. PRICE-S claims to deliver estimates to within 5% of actual cost after calibration by supplying industry-average values for actual input data that has not yet been specified. http://www.pricesystems.com/	• Parametric • CER • Life Cycle • Link to MS Office • Database
REVIC	REvised Intermediate COCOMO (REVIC) is available for downloading from the US Air Force Cost Analysis Agency (AFCAA). (Note: REVIC is not Y2K compliant and was replaced by SoftEST.) http://www.saffm.hq.af.mil/afcaa/	• Parametric • COCOMO

Product	Description	Features
SEAT	Software Estimation and Analysis Tool (SEAT) was a student project of Design Studio Group, East Tennessee State University, and appears to be unsupported. Available for download over the Internet, it is a Windows-based tool integrating Function Point Analysis (FPA) with the COCOMO model. http://www.cs.umt.edu/RTSL/SoftEng/Index.html	• Parametric • COCOMO • Function Point
SEER	Galorath Incorporated (also known as GA SEER™ Technologies) provides a comprehensive set of decision-support and production optimization tools to help manage product design and manufacturing operations. SEER-SEM is the parametric cost model and SEER-SSM estimates the expected size of software. They derive cost, schedule, labor and materials estimates by assessing the interaction and impact of product, organizational and even operational variables. http://www.galorath.com/	• Parametric • Link to MS Project • Database
SLIM-Estimate	Quantitative Software Management (QSM) offers their clients Software Life-cycle Management (SLIM) tools for software cost estimating (SLIM-Estimate), reliability modeling, schedule estimating, planning, tracking, and benchmarking. http://www.qsm.com/	• Parametric • Life Cycle • Database • Link to MS Office
SoftEST	SPAWAR Systems Engineering Process Office (SEPO) makes available the Software Estimation Model (SoftEST) developed by MCR Federal Inc. for the Air Force Cost Analysis Agency. SoftEST is the follow-on to the REVIC software estimation model. It is capable of varying development environment at CSCI level and supports complex projects developed with rapid prototyping, incremental, evolutionary, and spiral development methods. http://sepo.spawar.navy.mil/sepo/estimation.html	• COCOMO • Life Cycle

Commercially available cost estimation tools try to offer the user greater utility by packaging the parametric model with a user interface, database of completed projects, some way of estimating the size of the project, and/or context-sensitive help. For example, more than one vendor offers a package that incorporates Function Point Analysis as the front end to a COCOMO model. Another vendor offers over 30 distinct units of measure, but the user must supply the 'gearing factor' that converts the estimate unit to SLOC. Yet another vendor offers a Requirements Management front end to a parametric model.

Whatever features any tool may have, most parametric models are likely to have the COCOMO equations at the core. Because of this widespread influence, what follows is a brief discussion on recent developments at the University of Southern California (USC) COCOMO Project.

COCOMO II[3]

The newest version of the cost model, COCOMO II, is a screen-oriented, interactive software package that addresses 1990s and 2000s software development practices. A development of the program of research conducted by Dr. Barry Boehm at the USC Center for Software Engineering (CSE), it is an update of COCOMO model first published in 1981.[4]

[3] Adapted from the COCOMO II User's Manual.
[4] Barry Boehm, Software Engineering Economics, Prentice-Hall, 1981.

A key feature of COCOMO II is that it implements formulas in three stages to estimate the effort, schedule, and cost required for developing a software product. Stage 1 supports the estimation of prototyping or Applications Composition efforts (not currently available). Stage 2 supports estimation in the Early Design stage of a project, when less is known about the project's cost drivers. Stage 3 supports estimation in the Post-Architecture stage of a project.

Thus, a software project manager can develop models of projects both before and while the software is being developed in order to identify potential problems in resources, personnel, budgets, and schedules. COCOMO II provides a breakdown of effort and schedule into software life-cycle phases and activities from the original COCOMO manual. These are still reasonably valid for Waterfall model software projects, but need to be interpreted for non-waterfall projects.

Designed for estimating software development efforts, COCOMO does nothing to address either hardware or operations costs, which are significant components in overall data system costs. On the other hand, COCOMO II is designed for combination with other software packages. Herein lies its usefulness as a powerful estimating engine within a cost estimation tool.

Interestingly, it is in fact used by many of the commercial and proprietary cost estimation tools described in this report. Its relationships, algorithms and interfaces are publicly available, well defined, and parameterized. Thus, size estimation tools (e.g., analogy, case-based, or function point), project planning and control tools, risk analysis software, and project management tools can be combined with COCOMO II in a relatively straightforward manner.

Assessment of the Most Promising Tools

Early in the survey, criteria were developed for selecting tools to accurately predict development and sustaining engineering costs for distributed systems and systems reuse. The criteria for selection in this phase of the overall task to develop a cost estimation model were:

- Does it estimate life-cycle costs?
- Can it interface with external applications (e.g., Excel)?
- Does it consider COTS?
- Does it consider reuse?
- Is it easy to use?
- How does it handle new technology/technology enhancement?
- Where could it best be used in our study?

From the list of existing tools given in Table 1, the tools showing the most promise for further investigation are:

- ACEIT
- COCOMO II
- Construx Estimate
- COSMOS
- COSTAR
- Cost Xpert
- ESTIMATE Pro
- PRICE-S
- SEER-SEM
- SLIM-Estimate

As previously mentioned, each parametric cost estimation model covered in this survey employs one or more of three methodologies. Putnam methodology is based on the insight that efficiently run software projects follow well-defined patterns that can be modeled with a set of exponential equations. COCOMO II is a continuation of work begun by Dr. Barry Boehm at USC. Monte Carlo simulation models complex interactions in the face of uncertain estimating assumptions.

Designed for estimating software development costs, most of the tools did not address life cycle costs. ACEIT, PRICE, SEER and SLIM-Estimate all claimed to address life cycle costs, but there were wide variations in how these costs were addressed. Cost Xpert, for example, equated the 'lifecycle' to software development activities from design through the deployment phase. Neither operations nor maintenance costs were addressed.

Most tools did have some kind of interface to external applications such as Microsoft Excel. No tool, with the exception of COCOMO II, stood out as having been explicitly designed for such interfaces. Where an interface was available, it was used primarily for data input and output.

All of the cost estimation tools addressed COTS and reuse issues, except for ACEIT, which is really intended for a much broader scope of analysis than estimating software development. How well these issues were addressed varied considerably among the tools.

The most difficult package to use was ACEIT. All the others were relatively easy to use, and COSTAR had the simplest and most elegant interface. Construx Estimate deserves special mention for its display of the scatter plot and confidence intervals resulting from its built-in stochastic modeling. It also has the advantage of being available at no charge.

All of the tools were able to handle new technology or technology enhancement from the point of view of incorporating new development methods, which is critical for any package. Construx Estimate and SLIM-Estimate went a bit further by incorporating stochastic modeling. Actually, these tools use the same (SLIM) model. COCOMO II gets an honorable mention as a major university source of new technology for software cost estimation.

It should be noted that ANGEL, which is not on the short list above, deserves some attention in that part of our study concerning cost estimation by analogy. It was the only tool using cased-based reasoning (analogy) for cost estimation. Hardly more than a software engineering research project, this tool did not 'make the cut' in selecting tools for the next phase. However, it may be worthy of consideration from the point of view of cost estimation by analogy.

Availability and Usefulness of Existing Estimation Tools

Availability

As previously noted, commercial-off-the-shelf (COTS) cost estimation tools are widely available on the open market. Sources for these tools include university research programs, commercial entities, and government agencies. More and more frequently, COTS tools are being used for commercial purposes and carry prices for single user licenses up to $4,000 or more. Prices for corporate or site licenses are usually negotiable.

Some researchers observe that the most common method in practice is cost estimation by analogy. Other than ANGEL, as previously noted, COTS cost estimation tools are parametric models. Software tools for estimation by analogy are not available on the commercial market. However, non-algorithmic effort estimation techniques have been proposed in the literature on software engineering research. These techniques include: rule induction, fuzzy systems, regression trees, neural networks and case-based reasoning. Only the latter two are receiving serious attention from software effort researchers. Expect to hear more about them in the future.

Usefulness (or Not)

Parametric models provide a quick, relatively painless way of generating a software development cost estimate. It is certainly easier than the 'bottom-up' approach in which a system design is broken down into components whose development effort can then be estimated by an engineer. However, parametric models do have certain drawbacks. Here are a few:

- Parametric models are notoriously inaccurate, varying as much as 100% when compared with actual data. Even when calibrated to past experience, the variance could be 30%.
- Few models provide life cycle cost estimates that include sustaining engineering and operations costs. Most models address software development only.
- Models require size estimates (e.g., SLOC, Function Points) as the key input. Some tools offer other units of measure, which must be calibrated back to SLOC.
- Model parameters could number in the hundreds, although not all are used for any given project. These parameters, however, are no better than the historical data.
- Calibration of the parametric model requires historical cost, schedule, and productivity data of the organization doing the development.

Size Matters

Every parametric model without exception requires user input on the size of the software development project. Input may be either Source Lines of Code (SLOC), Function Points or other unit of measure. Function Point Analysis is an excellent alternative to SLOC for estimating the size of a project (see box). In fact, some COTS cost estimation tools are offered with a Function Point Analysis front end as part of the package. Finally, for reasons that are not altogether clear, some cost models input the effort (man-months, FTE, etc) required for development.

Function Points

Allen Albrecht at IBM developed what is called the Function Point methodology. This methodology is based on the premise that size of a software project can be estimated early, during requirements analysis, by counting the inputs and outputs of the system. Most tools convert the Unadjusted Function Point count to an equivalent number of SLOC, and use that in the COCOMO equations to make estimates. Five classes of items are counted:

- External Inputs
- External Outputs
- Logical Internal Files
- External Interface Files
- External Inquiries

Calibration

All cost estimating methods require a historical database for calibration. Even the 'bottoms-up' method of estimation depends on a database: the memory of the estimating engineer. As long as the environment, tools, methods, practices, and skills of the people do not change dramatically from one project to the next, historical data is a simple and useful calibration tool.

Boehm calibrated the COCOMO II model to 161 data points (projects) using a Bayesian statistical approach blending empirical data with expert opinion. It should be noted that if an organization calibrates the Effort Adjustment Factor in COCOMO II to its own empirical data, the accuracy of the model could be improved substantially over the results of generic calibration.

To make it easier for the user, many existing cost estimating tools come equipped with a proprietary 'industry standard' historical database (some with thousands of projects stored) or the COCOMO II database. Nevertheless, vendors of parametric models state that the most accurate models are those calibrated to the historical data of the developing organization.

In this regard, it should be noted that the study team is building a 'comparables' database from information obtained from NASA data centers. This database will be used for cost estimation by analogy and may also be used in parametric models, if the data collected matches that required by the cost estimation tool. (This is to be determined.)

Software Development Methods

New development methods change the way you size a project. Historically, parametric models have used the traditional Waterfall model of software and hardware development. But, how are open, extensible distributed data systems best modeled? Software development methods have evolved and today's cost estimation models must adapt to new technology. Table 2 shows some alternative software development methods that up-to-date cost estimation tools can handle. One of these alternatives may be a better match for the way that ESE data systems are developed.

Table 2. Software Development Methods

Method	Description
Evolutionary	Software requirements and design will change and grow throughout the development process. Often associated with user-oriented systems or systems not yet fully understood—high volatility.
Incremental	A linear model of the software development process that allows the software developer to iterate among the activities within each life cycle phase for each increment defined for the system.
Object Oriented	The use of all object-oriented techniques for requirements analysis, design, coding, and testing by a development team that is experienced and motivated to use object-oriented approaches.
Prototype	Informal development process applicable for prototypes, proof of concept, or demonstration software. Development is iterative, with minimal up front requirements effort.
Spiral	A cyclical model of the software development process where a repeating set of activities is performed on an increasingly more detailed representative of the product.
Waterfall	A linear model of the software development process where the activities of each phase of the life cycle must be completed before continuing to the next phase.

Industry Standards

One of the most useful aspects of the new crop of software estimation tools is the incorporation of modifiable parameters reflecting current industry standards and practices (e.g., SEI CMM, ISO 9001, etc). Vendors of some of these tools collect historical data from industry experience gained while operating under software engineering standards. This suggests another criterion, not previously considered, for selecting a cost estimation model for further study.

Estimating System Policies and Procedures

Calibrated and validated parametric estimating models and techniques satisfy all Government procurement regulations, provided that estimating system policies and procedures are well established and used consistently. Regulations define an "adequate estimating system" as being (1) established, maintained, reliable, and consistently applied; and (2) produces verifiable, supportable, and documented cost estimates. These requirements apply when parametric cost estimation tools are used to cost proposals.

Summary of Research Into Parameterizing Cost

Research into parameterizing cost for distributed systems operations and for user demand for distributed science data is notably absent from software engineering research efforts. Some advanced, non-commercial tools may have support for sophisticated modeling techniques such as system dynamic modeling or knowledge based modeling. Other researchers may take different approaches to cost estimation (e.g., requirements-based or architecture-based) worth noting. Three research efforts are offered here for consideration:

- User demand for distributed science data center
- Model-based architecting and software engineering
- Architectural approach to software cost modeling

User Demand for Distributed Science Data

Dr. Bruce Barkstrom is Head, Distributed Active Archive Center (DAAC) at NASA Langley Research Center. In two half-day meetings with the study team, he presented three models for our consideration:

- COTS cost estimation tool (Cost Xpert)
- Markov model for user demand
- Continuous data flow model

Barkstrom's presentation of the Cost Xpert cost estimation tool was useful in that it validated the study team's decision to place the product on the 'short list' for further consideration. Of greater significance to the cost estimation survey were his efforts to quantify data volumes and user access rates. In particular, the data on classes and numbers of users in the user demand model as

well as data volume could serve as a 'proxy' measures for the scope (size) of data system development efforts. This possibility remains to be explored.

The 'continuous data flow model' illustrates the variability of computational algorithms, which directly impacts software development efforts. In practice, data system requirements within the Earth Science Enterprise (ESE) are constantly changing to reflect new knowledge and shifting objectives. Thus, the traditional Waterfall model, which is but a single iteration of the software development process, is simply not appropriate for estimating costs in this case. Referring back to Table 2, it can be seen that there may be alternatives that are better suited. If so, then cost estimation is bound to be more accurate if it reflects real events.

The 'market research' approach to modeling user demand deserves additional consideration. While there is a dearth of research into user demand for distributed scientific data centers, the literature is replete with user models for a great many other products, such as access to the Internet, which may be quite useful. Most advanced market forecasting methods use methods such as exponential smoothing, time series analysis, multivariate regression, and artificial neural networks. Indeed, there are COTS tools for modeling user behavior, which should be reviewed for possible use in our application. (Market research tools are beyond the scope of this survey.)

Model-Based Architecting and Software Engineering

Alternative Models	
Product Models	• Architecture • Requirements • Code
Process Models	• Tasks • Activities • Milestones
Property Models	• Cost • Schedule • Performance • Dependability
Success Models	• Stakeholder win-win • Business case

Led by Dr. Barry Boehm, the University of Southern California Center for Software Engineering (USC-CSE) has been developing, applying and refining an approach called Model-Based Architecting and Software Engineering (MBASE). The claim is that in order to determine whether any software or system architecture is satisfactory, one needs considerably more than just the architecture. MBASE focuses on ensuring that a project's product, process, property, and success models (see inset) are consistent and mutually enforcing. Support for this research is provided by the USC-CSE Affiliates (more than 20 large corporations), Defense Advanced Research Projects Agency (DARPA), and the Federal Aviation Administration (FAA).

Architectural Approach to Software Cost Modeling

Design and development of the software architecture becomes more significant in a situation of increasing size and complexity. Research at the Software Engineering Institute (SEI) suggests an architectural approach to software cost modeling.[5] At SEI, a research topic of great interest is the stimulus/response characteristics of the software architecture. From the user point of view, these characteristics are the quality attributes of the architecture. Utility for the user translates to the quality attributes of performance, modifiability, availability, and security.

[5] Jai Asundi, Rick Kazman, and Mark Klein, An Architectural Approach to Software Cost Modeling, SEI Interactive, March 2000.

SEI has developed the Architecture Tradeoff Analysis Method[6] as a framework for reasoning about technical tradeoffs. This framework may be useful to the scientific community in its role of assessing the utility of data center designs. It is included here as an alternative to purely cost estimation efforts as the means for arriving at architectural choices.

Conclusion

Building a model for estimating the cost of distributed scientific data systems (and centers) is highly dependent on the software development environment, including methods and standards. The selected tool must not only have this capability, but must also fit the cost estimation process.

Figure 1 illustrates a (preliminary) concept for the cost estimation process. In that process, a parametric cost model is only one element of the overall picture. Modern cost estimation tools recognize this and are making provision for interfacing with a variety of tools related to the development effort. These tools include, for example, functional modeling (IDEF0, IDEF1X), requirements management (Easy Win/Win, DOORS, Rational Rose), software sizing, and project management (Microsoft Project).

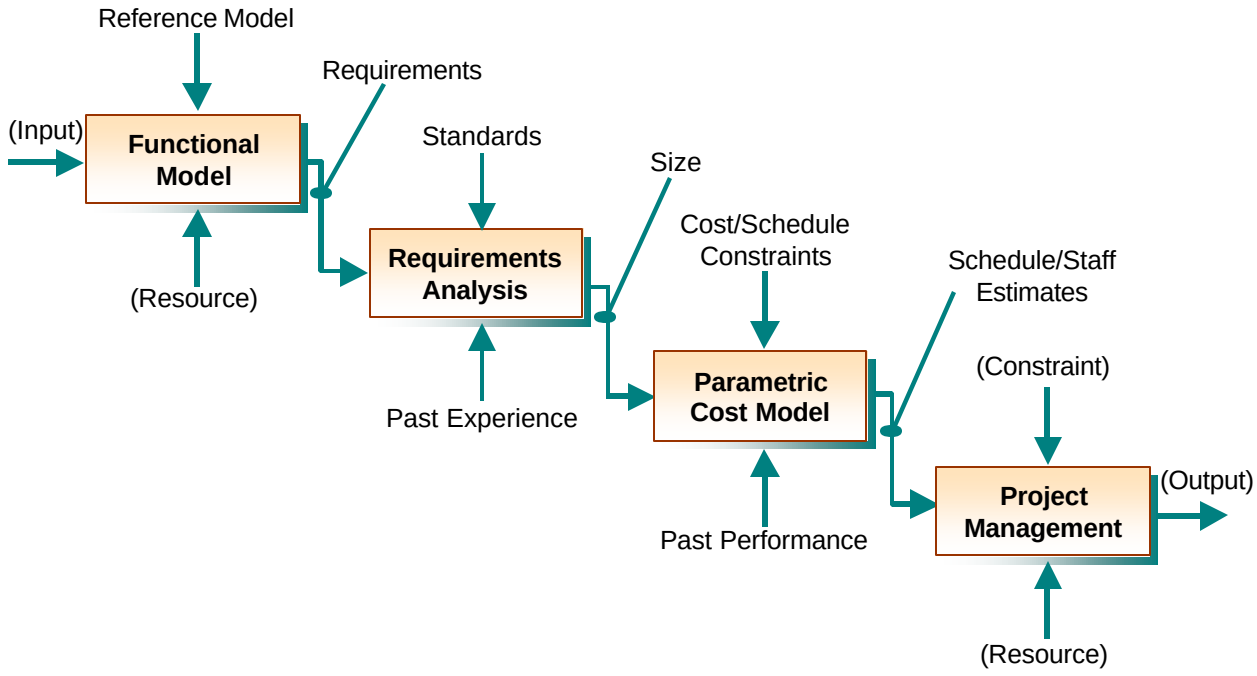

Figure 1. Cost Estimation Process Model (Preliminary)

[6] R. Kazman, M. Barbacci, M. Klein, S.J. Carriere, and S.G. Woods, "Experience with Performing Architecture Tradeoff Analysis," Proceedings of ICSE 21 (Los Angeles, CA), May 1999, 54-63.

Next Steps

The next step in this section of Task 34 is to construct a development cost model for SEEDS data centers, including the cost impact of software reuse, using one of the COTS cost estimation tools. In preparation for this step, the study team plans on completing the following near-term actions (by the end of February 2002):

- Finalize a short list of recommended cost estimating tools, with justification.
- Participate in the February 2002 SEEDS Community Workshop.
- Demonstrate proficiency in estimating costs using analogy and parametric models.
- Perform comparative and sensitivity analyses to identify optimum cost estimation tool(s).
- Survey user demand models with potential for supporting SEEDS cost estimation.